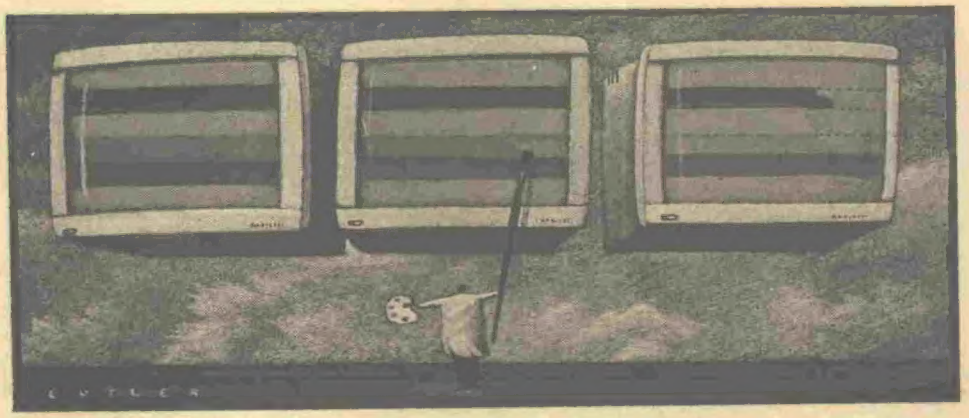
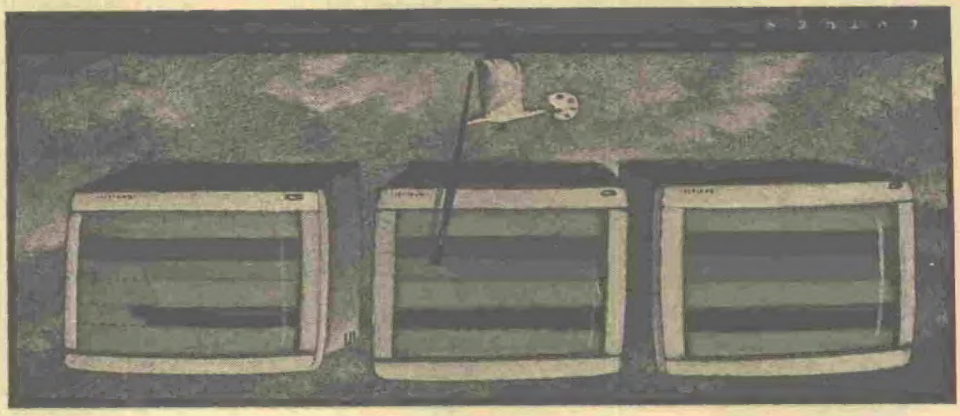
ชูเกียรติ แสงมณี และ ส.อ. บุญเรือน คงเขียว

การเขียนไดรเวอร์กราฟิก ภาษาไทย บนจอวีจีเอ

แนะนำเทคนิค วิธีการเขียนไดรเวอร์กราฟิก ภาษาไทยบนจอภาพ ทั้งภาษาปาสคาลและภาษาซี





มือผู้เขียนได้เขียนไดรเวอร์กราฟิกขึ้นมาในคราวที่แล้ว มีผู้สงสัยอยู่หลายท่านว่าทำไมต้องเขียนพร้อมกันทั้งภาษาปาสคาลและภาษาซี น่าจะแยกกันเขียนทีละภาษา เพราะดูแล้วทำให้อ่านง่ายกว่ากันเยอะเลย ผู้เขียนก็เลยขอแสดงความคิดเห็นสักเล็กน้อย หวังว่าจะไม่เสียเวลามากนัก

ประการแรก สำหรับผู้ที่เริ่มต้นที่จะเรียนภาษาคอมพิวเตอร์ จะได้ลองเปรียบเทียบดูลักษณะของตัวภาษาด้วยกันได้ ผู้เขียนเองก็พยายามเขียนโดยใช้คำสั่งง่าย ๆ ไม่ซับซ้อนในโครงสร้างของฟังก์ชัน (Function) หรือโพรซีเดอร์ (Procedure) ให้ใกล้เคียงกันมากที่สุด เมื่อผู้อ่านได้ติดตามคอลัมน์นี้ต่อไปจะเห็นถึงความแตกต่างของโปรแกรมเมื่อคอมพายล์แล้วทั้งสองภาษา ในเรื่องของความเร็ว หรือขนาดของไฟล์ หรือการจัดการเกี่ยวกับหน่วยความจำ และอาจจะทำให้ท่านผู้อ่านที่ลังเลอยู่กับการเลือกเรียนภาษาใดภาษาหนึ่งตัดสินใจได้ในที่สุด

ประการที่สอง เป็นเรื่องของผู้อ่านที่มีความเข้าใจในภาษาปาสคาลหรือภาษาซีอยู่แล้ว สามารถที่จะศึกษาเทคนิคและวิธีการต่าง ๆ ไปใช้งานได้เลย

ประการที่สาม ผู้เขียนเองฉนัดในการเขียนบทความที่ให้ผู้อ่านต้องมานั่งศึกษารายละเอียดของโปรแกรมเอาเองทั้งหมด ประโยชน์ที่จะได้รับอย่างน้อยก็จะทำให้ท่านผู้อ่านเกิดทักษะ และมีแนวความคิดของตนเองมากขึ้น เมื่อต้องอ่านหนังสือจากต่างประเทศจะพบว่าเขาไม่มานั่งอธิบายโปรแกรมทีละบรรทัดเพื่อความเข้าใจ เพราะการอธิบายแบบนี้จะทำให้เกิดการปลูกฝังเคยใช้อย่างไรก็ใช้อย่างนั้น ขาดความคิดริเริ่ม

นี่เป็นความคิดเห็นส่วนตัวของผู้เขียนเอง อาจจะไม่ตรงกับความคิดของหลาย ๆ ท่านก็เป็นได้ แต่เมื่อผู้เขียนอธิบายหลักการทั่วไปเพียงอย่างเดียว ทำให้สามารถรวบรวมรายละเอียดให้อยู่ในฉบับเดียวกันได้ และผลพลอยได้ที่น่าจะเป็นประโยชน์มากคือ ผู้เขียนพยายามหาวิธีการประกาศตัวแปรของ

ทั้งสองภาษาให้ใกล้เคียงกันมากที่สุด เมื่อภาษาปาสคาลต้องใช้ตัวแปรแบบ Constant แต่ในภาษาซีมีแต่การ Define ตัวแปรไว้ก่อน ผู้เขียนก็หันไปใช้ภาษา C++ เพื่อเปรียบเทียบการประกาศตัวแปรแบบ Constant (ภาษา C++ มีตัวแปรแบบ Constant) กับ Define

ถึงตอนนี้ผู้เขียนกับคุณหมอวิโรจน์ ลาภไพบูลพงศ์ หนึ่งในทีมงาน Rama Spell Check และเป็นหนึ่งในทีมงานพัฒนาด้านโปรแกรมที่ได้รวมทีมกันขึ้นมาเพื่อพัฒนาโปรแกรมคอมพิวเตอร์ในแนวความคิดใหม่ มีความคิดว่าผู้ใช้น่าจะพิจารณาเองว่าจะใช้อย่างไรกับโปรแกรมของเราเอง เพราะการใช้ Define เป็นการบอกว่าให้แทนที่ข้อความนี้ด้วยอะไร (Syntax) แต่การใช้ Constant จะกินเนื้อที่ในไฟล์ .EXE ของเรา แต่มีความแน่นอนกว่า สามารถบอกได้ว่าเป็น Constant ชนิด char, int หรืออื่น ๆ ได้ ในการคอมพายล์ด้วย Turbo C++ หรือ Borland C++ ผู้เชี่ยวชาญได้แนะนำไว้ว่าให้ใช้ Constant มากกว่า Define เพราะเวลาคอมพายย์จะได้ไม่สับสน โดยเฉพาะอย่างยิ่งในการวนรอบ (for, while เป็นต้น) เป็นคำแนะนำจาก MASTERING BORLAND C++ ของสำนักพิมพ์ SAMS ที่ผู้เขียนใช้ในการค้นคว้าและอ้างอิงในการเขียนบทความและเขียนโปรแกรม

ความละเอียดของจอภาพ

ก็ออกนอกเรื่องมาพอสมควรแล้ว กลับมาสู่เรื่องเดิมของเราอีกครั้ง และเตรียมพร้อมกับการทำความเข้าใจด้วยตนเองตามแบบฉบับต่อไป คราวที่แล้วได้กล่าวถึงไดรเวอร์กราฟิกบนจอภาพสีเดียวบนการ์ดเฮอร์คิวลิสซึ่งไม่มีอะไรยุ่งยากมากนัก ทำงานแบบตามหลักการของคอมพิวเตอร์เท่านั้น อาจมีเพียงการหาเส้นสแกนไลน์ที่อาจจะสร้างความสับสนอยู่บ้าง แต่ก็ไม่เกินความสามารถของผู้อ่านที่จะทำความเข้าใจได้ คราวนี้ก็เป็นเรื่องของจอสีที่มีหลายชนิดหลายแบบ และความละเอียดของจุดสีก็มีตั้งแต่ต่ำสุดจนถึง Super VGA ก่อนอื่นก็ขอให้ท่านผู้อ่านทำความเข้าใจกับจอภาพชนิดต่าง ๆ ให้ดีเสียก่อน สำหรับจอสีที่พบเห็นได้ในประเทศไทยก็มี เช่น จอ CGA (Color Graphic Adaptor), จอ MCGA (Multi Color Graphic Array), จอ PGA (Professional Graphic Adaptor) จอภาพเหล่านี้ (รวมถึง CRT Controller Card ด้วย) จะกล่าวถึงหลักการโดยทั่วไปเท่านั้น โดยจะกล่าวรวมกันในแบบสีมาตรฐาน EGA (Enhance Graphic Adaptor) และ VGA (Video Graphic Array) สำหรับรายละเอียดการแสดงผลแบบกราฟิก ดังในตารางที่ 1

ในตารางที่ 1 แสดงไว้เพื่อเปรียบเทียบให้เห็นถึงความแตกต่างระหว่างจอภาพ CGA, EGA และ VGA ทุก ๆ ความละเอียดและจำนวนสี จะเห็นได้ว่าจอภาพ VGA แสดงผลได้ครอบคลุมทั้งหมด ขณะที่จอภาพ EGA แสดงผลได้สูงสุดที่ 640 × 350 จุด (ต่อหนึ่งหน้าจอภาพไม่ใช่ต่อนิ้ว) ที่ 16 สี ส่วนจอภาพ CGA มีความละเอียดน้อยที่สุดและจะไม่กล่าวถึงในที่นี้ สำหรับช่องฟังก์ชันนั้นมีตั้งแต่ฟังก์ชัน 0 ถึง 13h . มีบางฟังก์ชันที่ข้ามไปเพราะเป็นการแสดงผลในรูปของตัวอักษร (Text Mode) เมื่อเราต้องการให้แสดงผลเป็นแบบกราฟิก เราจะใช้บริการของไบออส (BIOS = Basic Input Output System) โดยการเรียกอินเตอร์รัพต์ (Interrupt) หมายเลข 10h (Video Subsystem) และเลือกความละเอียดของจอภาพจากฟังก์ชันต่าง ๆ ได้โดยตรง ซึ่งต่างจากจอภาพแบบเฮอร์คิวลิสโดยสิ้นเชิง จากตารางที่ 1 จะเป็นการแสดงความละเอียดที่ไอบีเอ็มกำหนดมาตรฐานไว้เท่านั้น ส่วนที่เหลือนอกเหนือจากนี้จะเป็นส่วนของ Super VGA ซึ่งก็จะได้พูดถึงในคราวต่อ ๆ ไป

หมายเลขฟังก์ชัน(Hex)	ความละเอียด	จำนวนสี	จำนวนหน้า	จอภาพ
4	320 × 200	4	1	CGA, EGA, VGA
5	320 × 200	4, bw*	1	CGA, EGA, VGA
6	640 × 200	2, bw*	1	CGA, EGA, VGA
D	320 × 200	16	2,4,8**	EGA, VGA
E	640 × 200	16	1,2,4**	EGA, VGA
F	640 × 350	bw*	1,2**	EGA, VGA
10	640 × 350	16,64**	1,2**	EGA, VGA
11	640 × 480	2	1	VGA
12	640 × 480	16	1	VGA
13	320 × 200	256	1	VGA

หมายเหตุ * bw = ขาวดำ
** ขึ้นอยู่กับหน่วยความจำบน CRT Controller Card

ตารางที่ 1 รายละเอียดการแสดงผลแบบกราฟิกตามมาตรฐาน

รายละเอียดที่เกี่ยวกับค่าสี

เมื่อทราบรายละเอียดเกี่ยวกับความละเอียดของจอภาพชนิดต่าง ๆ เป็นที่เรียบร้อยแล้ว ก็มาดูรายละเอียดที่เกี่ยวกับค่าสีกันบ้างว่า ค่าสีเลขอะไรได้สีอะไร ดังแสดงในตารางที่ 2

จากในตารางที่ 2 เป็นการแสดงชื่อสีและค่าของสีในแบบตัวเลขที่เป็นมาตรฐาน 16 สี หากท่านใช้คอมพายเลอร์ของบอร์แลนด์และเรียกใช้ GRAPH. TPU หรือ GRAPHICS.H เราก็สามารถใช้ชื่อสีแทนค่าที่เป็นตัวเลขได้เลย ค่าที่เป็นตัวเลขนี้จะเป็นประโยชน์อย่างมากในการหาส่วนผสมของสี, แพลน (Plane) ที่ต้องใช้ แต่ผู้ใช้งานก็ไม่ต้องจดจำตัวเลขมากมาย ถ้าผู้ใช้งานจำหลักการของทฤษฎีสีแสงได้ เช่น IRGB (Intensity, Red, Green และ Blue เมื่อมองจากซ้ายไปขวาในตารางเลขฐานสองซึ่งต่างกับทฤษฎีสีระบาย (Paint) มี CMYB = Cyan, Magenta, Yellow และ Black ตรงนี้คงต้องช่วย ๆ จำไว้สักหน่อย เพราะท่านจะต้องนำมาใช้ในตอนที่ต้องการพิมพ์ภาพสี ส่วนค่าสี (Palette) เป็นค่าที่บอกตำแหน่งตารางสี ไม่ใช่ค่า Palette

ชื่อสี		ค่า (Hex)	ค่า (Binary)	ค่าสี (Palette)
ดำ	Black	0	0000	0
น้ำเงิน	Blue	1	0001	1
เขียว	Green	2	0010	2
ฟ้า	Cyan	3	0011	3
แดง	Red	4	0100	4
ม่วง	Magenta	5	0101	5
น้ำตาล	Brown	6	0110	6
เทา	Gray	7	0111	7
เทาเข้ม	DarkGray	8	1000	38
น้ำเงินอ่อน	LightBlue	9	1001	39
เขียวอ่อน	LightGreen	A	1010	3A
ฟ้าอ่อน	LightCyan	B	1011	3B
แดงอ่อน	LightRed	C	1100	3C
ม่วงอ่อน	LightMagenta	D	1101	3D
เหลือง	Yellow	E	1110	3E
ขาว	White	F	1111	3F

ตารางที่ 2 ค่าของสีต่าง ๆ ที่กำหนดไว้ (Default Values)

ส่วนของการผสมสี

เมื่อทราบค่าสีแล้ว ก็มาถึงในส่วนของการผสมสีกันบ้าง เนื่องจากสีต่าง ๆ ที่ได้ขึ้นมานั้นเกิดจากการนำเอาแม่สีมาผสมกัน เพื่อให้เกิดสีต่าง ๆ ขึ้นมา ซึ่งส่วนผสมของสีต่าง ๆ ก็ได้ตามตารางที่ 3

ในตารางที่ 3 มีส่วนที่น่าสนใจส่วนผสมของสี ใช้เป็นตัวเลือกแพลนเมื่อต้องการแสดงผลสีใด สำหรับค่าอัตราส่วนมีความสำคัญตอนที่ผู้ใช้ต้องการตั้งค่าระดับสี (Set Palette) อัตราส่วนสีมี 64 ระดับ เริ่มตั้งแต่ 0 ถึง 63 ระดับ (ไม่ว่าจะเป็น 16 สี หรือ 256 สี) และค่าที่ปรากฏในตารางที่ 3 นี้เป็นค่ามาตรฐานของ 16 สีแรก เมื่อคิดเป็นเปอร์เซ็นต์แล้วก็จะได้ออกมาในช่องถัดไปของตาราง ทำให้ง่ายต่อการคำนวณหาค่า Gray Scale ตามสูตรท้ายตาราง การตั้งค่าระดับสีและค่า Gray Scale มีประโยชน์และใช้งานอย่างไรนั้นเป็นเรื่องที่น่าสนใจมากในงานด้านกราฟิก ตอนนี้ขอเก็บไว้ก่อน จะอธิบายโดยละเอียดในตอนต่อไป

หน่วยความจำบนการ์ดสี

ก่อนจะเริ่มเข้าสู่ระบบกราฟิกสี ขอคุยกันถึงเรื่องหน่วยความจำบนการ์ดสีสักเล็กน้อย เพื่อจะได้เห็นถึงความแตกต่างระหว่างจอสีกับจอเดียว (Hercules) ได้ดีขึ้น โดยขอข้ามในโหมด 4 สี (CGA) มาเป็น 16 สี ในจอภาพแบบ EGA และ VGA หน่วยความจำของกราฟิกสีเริ่มต้นที่เซกเมนต์ (Segment) A000h ออฟเซต (Offset) ที่ 0000 หน้าที่ 1 (Page #1) เซกเมนต์ที่ A800h ออฟเซตที่ 0000 หน้าที่ 2 ในโหมดนี้เราจะใช้ข้อมูล 1 ไบต์ (Byte) ต่อข้อมูล 8 จุด (Pixel) ดังนั้นถ้าคิดอย่างง่าย ๆ หน่วยความจำที่ต้องใช้ในความละเอียดที่ 640×350 จุดต่อหนึ่งหน้า ที่หนึ่งสีเท่ากับ 32 กิโลไบต์ (Kbyte) แต่มีหน่วยความจำบางส่วนที่ไม่ได้ใช้รวมอยู่ด้วย ถ้าเป็น 16 สี คงต้องใช้หน่วยความจำถึง 512 กิโลไบต์ ซึ่งเป็นไปไม่ได้ เพราะนอกจากจะสิ้นเปลืองมากแล้ว ตำแหน่งของหน่วยความจำ (Address) นี้จะซ้ำกับตำแหน่งของฮาร์ดดิสก์และไบออส จึงต้องมีการกำหนดให้แบ่งการแสดงผลออกเป็นแพลน (Plane) ต่าง ๆ ได้ทั้งสิ้น 4 แพลน ทำให้ใช้หน่วยความจำเพียง 128

สี	ส่วนผสม	ค่าอัตราส่วน*			เปอร์เซ็นต์สี**		
		R	G	B	R	G	B
ดำ	ไม่มีส่วนผสม	0	0	0	0	0	0
น้ำเงิน	น้ำเงิน	0	0	42	0	0	66
เขียว	เขียว	0	42	0	0	66	0
ฟ้า	เขียว น้ำเงิน	0	42	42	0	66	66
แดง	แดง	42	0	0	66	0	0
ม่วง	แดง น้ำเงิน	42	0	42	66	0	66
น้ำตาล	แดง เขียว	42	21	0	66	33	0
เทา	แดง เขียว น้ำเงิน	42	42	42	66	66	66
เทาเข้ม	ความเข้ม	21	21	21	33	33	33
น้ำเงินอ่อน	ความเข้ม น้ำเงิน	21	21	63	33	33	100
เขียวอ่อน	ความเข้ม เขียว	21	63	21	33	100	33
ฟ้าอ่อน	ความเข้ม, เขียว, น้ำเงิน	21	63	63	33	100	100
แดงอ่อน	ความเข้ม แดง	63	21	21	100	33	33
ม่วงอ่อน	ความเข้ม แดง น้ำเงิน	63	21	63	100	33	100
เหลือง	ความเข้ม แดง เขียว	63	63	21	100	100	33
ขาว	ความเข้ม แดง เขียว น้ำเงิน	63	63	63	100	100	100

ตารางที่ 3
ส่วนผสมของแสงสีต่าง ๆ
ที่กำหนดไว้
(Default Values)

หมายเหตุ * เลขฐานสิบ
**ร้อยละ (Percentage)

Gray Scale = (0.30 × Red) + (0.59 × Green) + (0.11 × Blue)

กิโลไบต์ และให้แสดงผลพร้อมกันทั้ง 4 แพลนในเวลาเดียวกัน ดังนี้

- แพลนที่ 0 แสดงผลสีน้ำเงิน
- แพลนที่ 1 แสดงผลสีเขียว
- แพลนที่ 2 แสดงผลสีแดง
- แพลนที่ 3 แสดงผลความเข้ม (Intensity)

เมื่อเราต้องการให้จอภาพของเราแสดงผลสีใดก็ให้ดูจากตารางที่ 3 แล้วเราจะเห็นได้ว่าส่วนผสมของสีต่าง ๆ จะตรงกับค่าของเลขฐานสอง (Binary) ในตารางที่ 2 ซึ่งเป็นตัวชี้แพลนสีที่ต้องการแสดงผล เช่น สีม่วงอ่อน เกิดจากการผสมของความเข้มสีแดงและสีน้ำเงิน ตรงกับค่าเลขฐานสอง 1101 โดยที่ค่า 1 คือให้แพลนนั้น ๆ แสดงผล

สี	แพลนที่ 3	แพลนที่ 2	แพลนที่ 1	แพลนที่ 0
ม่วงอ่อน	1	1	0	1

ตารางที่ 4 แสดงผลของการผสมสีม่วงอ่อน

(บิตที่ 0, 2, 3) ค่า 0 บอกให้เรารู้ว่าไม่ได้ใช้งานแพลนนี้

ดังนั้นเราจึงสามารถกำหนดสีโดยเลือกแพลนก่อน แล้วจึงจะให้ค่าข้อมูลลงไปในตำแหน่งของหน่วยความจำอีกครั้งหนึ่ง เพื่อความเข้าใจที่ดี ก็ให้ดูตัวอย่างที่ 1

ตัวอย่างที่ 1

ให้แสดงผลตัวอักษร "A" สีม่วง ตำแหน่งที่ 0, 0

1: เลือกแพลนที่ 0
2: กำหนดตำแหน่งหน่วยความจำที่ A000:0000
3: ส่งข้อมูลของอักษร "A" ออกไป
4: เลือกแพลนที่ 2
5: กำหนดตำแหน่งหน่วยความจำที่ A000:0000
6: ส่งข้อมูลของอักษร "A" ออกไป
7: เลือกแพลนที่ 3

8: กำหนดตำแหน่งหน่วยความจำที่ A000:0000

9: ส่งข้อมูลของอักษร "A" ออกไป

ด้วยวิธีการเช่นเดียวกันนี้เราสามารถเลือกแพลนได้โดยตรงทั้ง 16 สี (0-F) แต่ไม่เป็นแบบมาตรฐานทั่วไป ขึ้นอยู่กับ CRT Controller Card และหน่วยความจำบนการ์ดด้วย (ไม่แนะนำให้ใช้วิธีนี้สำหรับการเขียนโปรแกรมใช้งานทั่วไป แต่ถ้าเราเขียนขึ้นใช้เฉพาะงานก็ไม่ว่ากัน)

ตัวอย่างที่ 2

ให้แสดงผลตัวอักษร "A" สีม่วงตำแหน่งที่ 0, 0

1: ให้รีจิสเตอร์บอกหมายเลขแพลนที่ใช้ทั้งหมด = Dh

2: กำหนดตำแหน่งหน่วยความจำที่ A000:0000

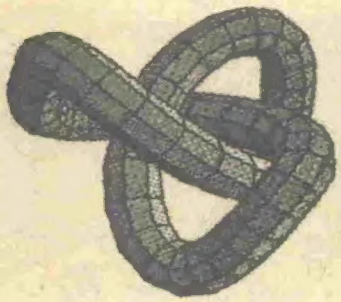
3: ส่งข้อมูลของอักษร "A" ออกไป

รีจิสเตอร์ ตัวบอกการสั่งงาน

ในการใช้งานตามตัวอย่างที่กล่าวมาแล้ว เราต้องใช้รีจิสเตอร์ (Register) เพื่อเป็นตัวบอกการสั่งงานสำหรับแพลนที่เราจะใช้ และรีจิสเตอร์นี้มีอยู่ 2 กลุ่ม คือ Sequencer Registers และ Graphics Controller Registers ส่วนอีก 3 กลุ่มที่ไม่ได้ใช้ในตอนนี้คือ General or External Registers, CRT Controller Registers และ Attribute Registers แต่ในโอกาสต่อไปเราต้องได้ใช้งานแน่นอน ถ้าท่านไม่เบื่อที่จะอ่านเสียก่อน

Sequencer Registers

สำหรับกลุ่มของ Sequencer Registers ก็จะประกอบไปด้วย



- Address
- Reset
- Clocking Mode
- Map Mask
- Character Map Select
- Memory Mode

Graphics Controller Registers

สำหรับกลุ่มของ Graphics Controller Registers ก็จะประกอบไปด้วย

- Graphics 1 Position
- Graphics 2 Position
- Graphics Address
- Set/Reset
- Enable Set/Reset
- Color Compare
- Data Rotate
- Read Map Select
- Mode
- Miscellaneous
- Color Don't Care
- Bit Mask

ล้างหน่วยความจำของจอภาพ

เรายังคงใช้บริการของไบออสเพื่อทำการอินเตอร์รัพต์หมายเลข 10h อยู่ ฟังก์ชัน 10h ใช้ในการกำหนดการแสดงผลที่ความละเอียด 640 × 350 จุด 16 สี และใช้ตัวหนังสือ (Font) ขนาด 8 × 14 จุดเหมือนเดิม แต่การเรียกใช้การแสดงผลแบบกราฟิก จะเป็นการเรียกอินเตอร์รัพต์เพียงอย่างเดียวไม่ได้ ต้องตั้งค่าต่าง ๆ ผ่านทางพอร์ต (Port) อย่างเช่น จอสีเดียว (Hercules) ทุกครั้งที่เรียกอินเตอร์รัพต์หมายเลข 10h เครื่องก็จะทำการล้างหน่วยความจำของจอภาพให้ (ยกเว้นการเลือก Bank ของหน่วยความจำที่มีทางเลือกให้ว่าจะล้างหน่วยความจำด้วยหรือไม่ ติดตามรายละเอียดในหัวข้อ Super VGA ต่อไป) ซึ่งต่างกับการแสดงผลแบบกราฟิกของจอสีเดียว (Hercules) ที่เราจะต้องล้างหน่วยความจำของจอภาพเองสำหรับความยุ่งยากที่เราต้องพบเห็นก็คงจะได้แก่ การกำหนดให้แสดงสีและเก็บค่าสีจากตำแหน่งต่าง ๆ บนจอภาพ

เริ่มเข้าสู่การแสดงผลแบบกราฟิก

คราวนี้เราคงพร้อมที่จะใช้งานการแสดงผลแบบกราฟิกกันแล้ว เริ่มต้นจากการเก็บค่าของสแกนไลน์ทั้ง 350 เส้นไว้ในอะเรย์ เพื่อความสะดวกในการเรียกใช้ และไม่ต้องคำนวณใหม่ทุกครั้งที่ต้องการหาค่าสแกนไลน์ สูตรในการหาค่าแต่ละเส้นมีดังนี้

```
Scan Line ที่ N = N × 80
```

เนื่องจากเราใช้ 640 จุดใน 1 เส้น และใช้ข้อมูล 1 ไบต์แสดงผล 8 จุด ดังนั้นใน 1 เส้นเราจึงใช้เพียง 80 ไบต์ จะเห็นได้ว่าค่าของสแกนไลน์ของจอสี EGA และ VGA จะเป็นการสแกนแบบเส้นต่อเส้นไปเรื่อย ๆ ซึ่งต่างกับ

จอภาพแบบเฮอร์คิวลิสและ CGA โดยที่จอภาพแบบเฮอร์คิวลิสมีการสแกนแบบข้ามหน่วยความจำทีละ 2000h คือ 0000h, 2000h, 4000h และ 6000h แล้วกลับมาที่ 0000h อีก หรือจะเรียกว่าเป็นการสแกนแบบทีละ 4 เส้น ส่วนจอภาพแบบ CGA มีการสแกนแบบข้ามหน่วยความจำทีละ 2000h คือ 0000h และ 2000h แล้วกลับมาที่ 0000h อีก หรือจะเรียกว่าเป็นการสแกนเส้นคู่และเส้นคี่นั่นเอง (ถ้าอ่านแล้วงง ๆ ก็กลับไปอ่านการทำไดรเวอร์กราฟิกสำหรับจอภาพแบบเฮอร์คิวลิสจากฉบับก่อนนี้ได้)

การเลือกแพลนสีเราใช้รีจิสเตอร์กลุ่ม Sequencer Registers ประเภท Map Mask 2 ตัวคือ อินเด็กซ์รีจิสเตอร์ (Index Register) ทางพอร์ต (Port) หมายเลข 3C4h และรีจิสเตอร์ข้อมูล (Data Register) ทางพอร์ตหมายเลข 3C5h ทุกครั้งที่ต้องการใช้แพลนสีใด เช่น เราต้องการให้แสดงผลตัว "A" เป็นสีเหลืองในตำแหน่งที่ 0, 0 ให้กำหนดได้ดังวิธีที่ 1 และวิธีที่ 2

วิธีที่ 1

ค่าของสีเหลืองเท่ากับ 1110 (ฐาน 2) ให้เลือกแพลนสีที่เป็นจริง (มีค่าเป็น 1) โดยมองจากขวามาซ้าย (ดูตารางที่ 2 ประกอบ)

1: กำหนดค่าพอร์ต 3C4h เท่ากับ 2 และพอร์ต 3C5h เท่ากับ 2 (แพลนที่ 1)

2: กำหนดตำแหน่งหน่วยความจำที่ A000:0000

3: ส่งข้อมูลของอักษร "A" ออกไป (หนึ่งตัวอักษรมี 14 ไบต์)

4: กำหนดค่าพอร์ต 3C4h เท่ากับ 2 และพอร์ต 3C5h เท่ากับ 4 (แพลนที่ 2)

5: กำหนดตำแหน่งหน่วยความจำที่ A000:0000

6: ส่งข้อมูลของอักษร "A" ออกไป

7: กำหนดค่าพอร์ต 3C4h เท่ากับ 2 และพอร์ต 3C5h เท่ากับ 8 (แพลนที่ 3)

8: กำหนดตำแหน่งหน่วยความจำที่ A000:0000

9: ส่งข้อมูลของอักษร "A" ออกไป

วิธีที่ 2

ค่าของสีเหลืองเท่ากับ 14 (ฐาน 10)

1: กำหนดค่าพอร์ต 3C4h เท่ากับ 2 และพอร์ต 3C5h เท่ากับ Eh

2: กำหนดตำแหน่งหน่วยความจำที่ A000:0000

3: ส่งข้อมูลของอักษร "A" ออกไป (หนึ่งตัวอักษรมี 14 ไบต์)

หมายเหตุ

เราให้ค่าของรีจิสเตอร์ทางพอร์ต 3C4h เท่ากับ 2 เสมอ เพื่อบอกให้รีจิสเตอร์ทางพอร์ต 3C5h รอรับหมายเลขแพลน เพราะพอร์ต 3C5h ทำงานอย่างอื่นได้ด้วย

ก่อนจาก

จากที่กล่าวมาแล้วเป็นเพียงการเขียนข้อมูลลงในหน่วยความจำของจอภาพ (Display Memory) เท่านั้น ถ้าหากเราต้องการอ่านข้อมูลจากหน่วยความจำของจอภาพแล้ว โดยวิธีนี้เราไม่สามารถนำมาใช้ได้ เพราะข้อมูลที่อ่านมาได้จะไม่ตรงกับความจริง เราจึงต้องใช้กลุ่มรีจิสเตอร์ Graphic Controller Registers เข้ามาช่วยเลือกแพลนที่ต้องการอ่าน ซึ่งรีจิสเตอร์กลุ่มนี้ใช้ง่าย สะดวก และปลอดภัย

ในคราวหน้าเราจะนำรายละเอียดและวิธีการอ่านข้อมูลจากหน่วยความจำของจอภาพมาลุยกันต่อ พร้อมทั้งนำเอาเทคนิคที่อธิบายมาแล้วมาใช้กับตัวอักษร (Font) ของซียูไรเตอร์ซึ่งมีขนาด 8 × 20 จุดต่อตัวอักษร และเทคนิคนี้สามารถใช้กับการแสดงภาพกราฟิกในรูปแบบต่าง ๆ ได้อย่างไร ผู้เขียนขอติดค้างไว้ 3 หัวข้อในคราวต่อไป แต่ตอนนี้ก็ให้ผู้อ่านได้ศึกษาโปรแกรมตัวอย่างที่อธิบายมาทั้งหมดก่อน และทำความเข้าใจไปด้วย ถนัดภาษาอะไร ก็ศึกษาภาษานั้น หรือถ้าต้องการจะศึกษาทั้งสองภาษาก็ไม่ว่ากัน

ที่ 1 แสดงโปรแกรม Graphic Driver ในภาษาซี

```
/**********************************
 *  Program Graphic Driver        *
 *  Use Turbo C++, Borland C++    *
 *  Chukiat Sangmanee             *
 **********************************/
#include <dos.h>
#include <stdio.h>

typedef unsigned char byte;
typedef byte  Gen_Map[256][14];
const Black             = 0,    /* Turbo C          */
      Blue              = 1,    /* #define Black 0 */
      Green             = 2,    /* #define Blue  1  */
      Cyan,             = 3,    /* .....            */
      Red               = 4,
      Magenta           = 5,
      Brown             = 6,
      Gray              = 7,
      DarkGray          = 8,
      LightBlue         = 9,
      LightGreen        = 10,
      LightCyan         = 11,
      LightRed          = 12,
      LightMagenta      = 13,
      Yellow            = 14,
      White             = 15;

Gen_Map Bit_Normal;
byte OldMode;
unsigned int Off_Line[350],G_Row=0,G_Col=0;

int GetVideoMode();
byte GPreadkey();
char KeyHit();
void Set_Color(byte Colour);
void G_Clear();
void OpenGraph();
void CloseGraph();
void G_Blink(byte TE);
void G_gotoxy(byte X,byte Y);
void G_write(byte *ThaiStr);

OpenFont(char *fi,byte *ThaiFont)
{
 FILE *Fp;
 if((Fp=fopen(fi,"rb"))==NULL)
    {
     printf("\n ****** File %s not found  *******",fi);
     return -1;
    }
    fread(&ThaiFont[0],sizeof(Gen_Map),1,Fp); fclose(Fp);
    return 0;
}

int GetVideoMode()
{
 union REGS r;
    r.h.ah=0x000F;
    r.h.al=0x0000;
    int86(0x0010,&r,&r);
    return(r.h.al);
}

void G_Clear()
{
 unsigned int Count,Xcount;
 for (Xcount=0;Xcount<350;Xcount++)
    for (Count=0;Count<80;Count++)
       pokeb(0xA000,Off_Line[Xcount]+Count,0x00);
}

void OpenGraph()
{
 union REGS r;
 int Count;
    for (Count=0;Count<350;Count++)
    Off_Line[Count]=Count*80;
    OpenFont("8X14.FON",*Bit_Normal); r.h.ah=0;
    r.h.al=0x0010;
    int86(0x0010,&r,&r);
}

void CloseGraph()
{
 union REGS r;
 r.h.ah=0;
 r.h.al=OldMode;
 int86(0x0010,&r,&r);
}

char KeyHit()
{
 union REGS r;
 r.h.ah=0x000B;
 int86(0x0021,&r,&r);
 if (r.h.al==0)
    return 0;
 else
    return 1;
}
```

รูปที่ 1 (ต่อ)

```
byte GPreadkey()
{
 union REGS r;
    r.h.ah=0;
    int86(0x0016,&r,&r);
    return(r.h.al);
}

void LineBlink(byte ST,byte SP,byte Tround)
{
 byte Count;
 for (Count=ST;Count<=SP;Count++)
    pokeb(0xA000,Off_Line[G_Row+Count]+G_Col,Tround);
    Count=0;
    while (!KeyHit() && (Count<100)) Count++;
}

void G_Blink(byte TE)
{
 byte ST,SP;
    if (TE==0)
      { ST=11; SP=12;}
    else
      { ST=3;  SP=11;}
 while (!KeyHit())
 {
    LineBlink(ST,SP,0x00FF);
    LineBlink(ST,SP,0x0000);
 }
void Set_Color(byte Colour)
{
 outportb(0x03C4,2);
 outportb(0x03C5,Colour);
}

void DispChar(byte ST,byte SP,byte *One_Char)
{
 byte Count;
 for (Count=ST;Count<=SP;Count++)
    pokeb(0xA000,Off_Line[G_Row+Count]+G_Col,One_Char[Count]);

}

void G_gotoxy(byte X,byte Y)
{
 if (X>80) X=80;
 if (Y>25) Y=25;
 G_Col=X-1;
 G_Row=(Y-1)*14; /* Charecter Hight */
}
```

รูปที่ 1 (ต่อ)

```
void G_write(byte *ThaiStr)
{
 byte ST,SP;
    while (*ThaiStr!=NULL)
      {
       if (*ThaiStr>31)
       switch (*ThaiStr)
       {
         case 216: case 217: case 218:{
           ST=12; SP=13;} break;
         case 209: case 212: case 213: case 214: case 215: case 231:
         { ST=2; SP=4;} break;
         case 232: case 233: case 234: case 235: case 236: case 237:
         { ST=0; SP=2;} break;
         default:{
           ST=0; SP=13; G_Col++; }
       }
       DispChar(ST,SP,Bit_Normal[*ThaiStr]);
       *ThaiStr++;
      }
```

รูปที่ 2 แสดงยูนิต Init_Col;

```
unit Init_Col;
interface
uses dos;
const Black                 = 0;
          Blue              = 1;
          Green             = 2;
          Cyan              = 3;
          Red               = 4;
          Magenta           = 5;
          Brown             = 6;
          Gray              = 7;
          DarkGray          = 8;
          LightBlue         = 9;
          LightGreen        = 10;
          LightCyan         = 11;
          LightRed          = 12;
          LightMagenta      = 13;
          Yellow            = 14;
          White             = 15;
type ByteImage=array[0..13] of byte;
var  Bit_Normal:array[0..255] of ByteImage;
          OldMode,ChHi,ChLow,StarPut,CharColour:byte
          G_Col,G_Row:word;
          Off_Line:array [0..349] of word;
```

ที่ 2 (ต่อ)

```
          Regs:registers;
          StartLine,StopLine:byte;
procedure Set_Colour(Colour:byte);
procedure G_gotoxy(X,Y:byte);
procedure G_Blink(TE:byte);
procedure G_write(ThaiStr:string);
procedure OpenGraph;
procedure CloseGraph;
procedure G_Clear;
function  G_readkey:char;
function  KeyHit:boolean;

implementation

procedure OpenFont;
var Fi:file;
begin
 assign(Fi,'8x14.fon');
 reset(Fi,1);
 blockread(Fi,Bit_Normal,sizeof(Bit_Normal));
 close(Fi);
end;
procedure Take_Off;
var St:word;
begin
 for St:=0 to 349 do
    Off_Line[St]:=St*80;
end;

function GetVideoMode:byte;
begin
 Regs.AH:=$0F;
 Regs.AL:=0;
 intr($10,Regs);
 GetVideoMode:=Regs.AL;
end;

procedure G_Clear;
var Count:integer;
        P:pointer;
begin
 P:=ptr($A000,$0000);
 fillchar(P^,32767,0);
end;

procedure OpenGraph;
var Count:byte;
begin
 Regs.AH:=0;
 Regs.AL:=$10;
 intr($10,Regs);
 G_Clear;
end;

procedure CloseGraph;
begin
 Regs.AH:=0;
 Regs.AL:=OldMode;
 intr($10,Regs);
end;

function KeyHit:boolean;
begin
 Regs.AH:=$0B;
 intr($21,Regs);
 if Regs.AL=0 then
    KeyHit:=false
 else
    KeyHit:=true;
end;

function G_readkey:char;
begin
    Regs.AH:=0;
    intr($16,Regs);
    ChLow:=Regs.AL;
    ChHi :=Regs.AH;
    G_readkey:=chr(ChLow);
end;

procedure G_Blink(TE:byte);
var St,Sp:byte;
 procedure LineBlink(Tround:byte);
 var Count:word;
 begin
    for Count:=St to Sp do
    mem[$A000:Off_Line[G_Row+Count]+G_Col]:=Tround;
    Count:=0;
    while not(KeyHit) and (Count<200) do inc(Count);
 end;

begin
 if TE=0 then
    begin St:=11; Sp:=12; end
 else
    begin St:=3; Sp:=11; end;
 while not(KeyHit) do
    begin
      LineBlink($FF);
      LineBlink($00);
    end;
end;
```

รูปที่ 2 (ต่อ)

```
procedure Set_Colour(Colour:byte);
begin
 port[$3C4]:=2;
 port[$3C5]:=Colour;
end;

procedure DispChar(G_Col,G_Row:word;One_Bitmap:ByteImage);
var Count:byte;
begin
    for Count:=StartLine to StopLine do
    mem[$A000:Off_Line[G_Row+Count]+G_Col]:=One_Bitmap[Count];
end;

procedure G_gotoxy(X,Y:byte);
begin
 if X>80 then X:=80;
 if Y>25 then Y:=25;
 G_Col:=X-1;
 G_Row:=(Y-1)*14; { CharHight }
end;

procedure G_write(ThaiStr:string);
var   Thai,Count,StrLength:byte;
       One_Image     :ByteImage;
begin
 if ThaiStr='' then exit;
 StrLength:=length(ThaiStr);
 Count:=1;
 while Count<=StrLength do
 begin
   Thai:=ord(ThaiStr[Count]);
   One_Image:=Bit_Normal[Thai];
    case Thai of
     0..22,24..31:;
     216..218:begin
                 StartLine:=12;
                 StopLine:=13;
               end;
     209,212..215,231:begin
                 StartLine:=2;
                 StopLine:=4;
               end;
     232..237:begin
                 StartLine:=0;
                 StopLine:=2;
               end;
    else
     begin
       StartLine:=0;
       StopLine:=13;
       inc(G_Col);
     end;
```

รูปที่ 2 (ต่อ)

```
        end;
        DispChar(G_Col,G_Row,One_Image);
        inc(Count);
  end;
 end;

 begin
  OpenFont;
  OldMode:=GetVideoMode;
  Take_Off;
  G_Row:=0;
  G_Col:=0;
  CharColour:=White;
 end.
```

รูปที่ 3 โปรแกรมสำหรับเมนูหลัก

```
#include "init_col.c"
void DispTable();
void main()
{
 byte Ch,Count,
              Head1[36]={"โปรแกรมติดตั้ง Alonggod Version 2.5"},
              Head2[20]={"1.ติดตั้งโปรแกรมที่"},
              Head3[19]={"2.ติดตั้งรูปภาพที่"},
              Head4[21]={"3.ติดตั้งตัวอย่างที่"},
              Head5[7] ={"4.ชื่อ"},
              Set1[12] ={"C:\\ALONGGOD"},
              Set2[20] ={"C:\\ALONGGOD\\PICTURE"},
              Set3[19] ={"C:\\ALONGGOD\\SAMPLE"},
              User_Name[19]={"ชูเกียรติ  แสงมณี"},
              Sele1[41]={"ถ้าต้องการเปลี่ยนแปลง โปรดเลือกตามหัวข้อ"};
 OldMode=GetVideoMode();
 OpenGraph();
 Set_Color(LightGreen);
 DispTable();
 Set_Color(Yellow);
    G_gotoxy(26,3);    G_write(Head1);
    G_gotoxy(20,8);    G_write(Head2);
    G_gotoxy(20,10);  G_write(Head3);
    G_gotoxy(20,12);  G_write(Head4);
    G_gotoxy(20,14);  G_write(Head5);
    G_gotoxy(36,8);    G_write(Set1);
    G_gotoxy(36,10);  G_write(Set2);
    G_gotoxy(36,12);  G_write(Set3);
    G_gotoxy(25,14);  G_write(User_Name);
 Set_Color(Green);
    G_gotoxy(26,17);  G_write(Sele1);
 Set_Color(LightMagenta);
```

ปที่ 3 (ต่อ)

```
do{
   G_gotoxy(31,22);  G_Blink(1);
   Ch=GPreadkey();
   G_gotoxy(29,22);  G_write(&Ch);
} while(Ch!=27);
CloseGraph();
}

void DispTable()
{
 byte Count, T1[58]={"----------"},

              T2[2] ={" "},

              T3[58]={"----------"},

              T4[58]={"----------"},

              T5[2] ={"L"};

   G_gotoxy(12,1); G_write(T1);
   for (Count=2;Count<18;Count++)
     {
       G_gotoxy(12,Count); G_write(T2);
       G_gotoxy(68,Count); G_write(T2);
     }
   G_gotoxy(12,5);  G_write(T3);
   G_gotoxy(12,16); G_write(T3);
   G_gotoxy(12,18); G_write(T4);
   G_gotoxy(30,20);
   Set_Color(Cyan);
   for (Count=0;Count<20;Count++) G_write(T5);
}
```

■

รูปที่ 4 (ต่อ)

```
G_write('----------');
for Count:=2 to 17 do
   begin
     G_gotoxy(12,Count); G_write(' ');
     G_gotoxy(68,Count); G_Write(' ');
   end;
G_gotoxy(12,5);
G_write('----------
');
G_gotoxy(12,16);
G_write('----------
G_gotoxy(12,18);
G_write('----------');
G_gotoxy(30,20);
Set_Colour(LightRed);
for Count:=0 to 19 do G_write(' ');
Set_Colour(Yellow);
G_gotoxy(26,3)    G_write('โปรแกรมติดตั้ง Alonggod Version 2.5');
G_gotoxy(20,8);   G_write('1.ติดตั้งโปรแกรมที่');
G_gotoxy(20,10);  G_write('2.ติดตั้งรูปภาพที่');
G_gotoxy(20,12);  G_write('3.ติดตั้งตัวอย่างที่');
G_gotoxy(20,14);  G_write('4.ชื่อ');
G_gotoxy(36,8);   G_write(Set1);
G_gotoxy(36,10);  G_write(Set2);
G_gotoxy(36,12);  G_write(Set3);
G_gotoxy(25,14);  G_write(User_Name);
G_gotoxy(26,17);  G_write('ถ้าต้องการเปลี่ยนแปลง โปรดเลือกตามหัวข้อ');
repeat
   G_gotoxy(31,22);  G_Blink(1);
   Ch:=G_readkey;
   G_gotoxy(29,22);  G_write(Ch);
until Ch=#27;
G_Clear;

CloseGraph;
end.
```

■

รูปที่ 4 โปรแกรมเมนูการติดตั้ง

```
uses Init_Col;
var Count:byte;
        Set1,Set2,Set3:string[29];
        User_Name:string[40];
        Ch:char;
begin
 OpenGraph;
 Set1:='C:\ALONGGOD';
 Set2:='C:\ALONGGOD\PICTURE';
 Set3:='C:\ALONGGOD\SAMPLE';
 User_Name:='ชูเกียรติ  แสงมณี';
 Set_Colour(Green);
 G_gotoxy(12,1);
```